In the Claims

1-2. (cancelled)

3. (previously presented) A method of enabling dynamic optimization of a computer program, comprising:

generating annotation information about said computer program, said annotation information being derived from information held by a compiler about references to individual memory locations; and

storing said annotation information with said computer program, said annotation information enabling a dynamic optimizer to optimize said computer program during execution.

4. (original) The method of claim 3, wherein generating annotation information comprises generating annotation information enabling replacement of subroutine calls with inline program code in said computer program while said computer program is being executed.

5. (original) The method of claim 3, wherein generating annotation information comprises a compiler generating said annotation information.

6. (original) The method of claim 3, wherein said computer program comprises at least one executable file.

7. (original) The method of claim 3, wherein said computer program comprises at least one source file.

8. (original) The method of claim 3, wherein said generating annotation information comprises generating

annotation information derived from runtime architecture and software conventions.

9. (cancelled)

10. (original) The method of claim 3, wherein said generating annotation information comprises generating annotation information identifying a unique stack pointer register to be used by said computer program.

11. (original) The method of claim 3, wherein said generating annotation information comprises generating annotation information comprising a list of non-ambiguous memory locations.

12. (original) The method of claim 11, wherein said annotation information enables said dynamic optimizer to obtain canonical names for said non-ambiguous memory locations.

13. (original) The method of claim 11, wherein said non-ambiguous memory locations comprise stack frame locations.

14. (original) The method of claim 3, wherein said generating annotation information comprises generating annotation information comprising a mapping of memory references to all non-ambiguous locations which are referenced.

15. (original) The method of claim 3, wherein said generating annotation information comprises generating annotation information comprising a list of canonical names of stack frame locations that are promotable.

16. (original) The method of claim 3, wherein said generating annotation information comprises generating annotation information comprising a guarantee that no stack frame location is live beyond the scope of the stack frame.

17. (original) The method of claim 3, wherein said generating annotation information comprises generating annotation information comprising a format and a location of stack unwinding information.

18. (currently amended) A method of dynamically optimizing a computer program, comprising:

reading annotation information derived from runtime architecture and software conventions used to compile said computer program, <u>said annotation information also being derived from information held by a compiler about references to individual memory locations,</u> said annotation information being stored with said computer program; and

dynamically optimizing said computer program based on said annotation information while said computer program is being executed.

19. (original) The method of claim 18, wherein said dynamically optimizing said computer program comprises a binary translator optimizing said computer program.

20. (original) The method of claim 18, wherein said dynamically optimizing said computer program comprises replacing subroutine calls in said computer program with inline program code.

21. (original) The method of claim 18, wherein said

dynamically optimizing said computer program comprises removing redundant callee-save register restores.

22. (original) The method of claim 18, wherein said dynamically optimizing said computer program comprises propagating constant arguments within said computer program.

23. (original) The method of claim 18, wherein said dynamically optimizing said computer program comprises promoting local data from a stack frame location to a register.

24. (original) The method of claim 18, wherein said dynamically optimizing said computer program comprises removing redundant callee register saves.

25. (original) The method of claim 18, wherein said dynamically optimizing said computer program comprises removing stack frame allocation.

26. (previously presented) Apparatus for enabling dynamic optimization of a computer program, the apparatus comprising:

one or more computer readable storage media; and

computer executable instructions stored in the one or more computer readable storage media, the computer executable instructions comprising:

instructions for generating annotation information about said computer program, wherein said annotation information enables a dynamic optimizer to optimize said computer program during execution, said annotation information being derived from information held by a compiler about references to individual memory locations; and

instructions for storing said annotation information with said computer program.

27. (cancelled)